

A Survey of Robust LiDAR-based 3D Object Detection Methods for Autonomous Driving

Walter Zimmer¹, Emeç Erçelik¹, Xingcheng Zhou¹,
Xavier Jair Diaz Ortiz¹, and Alois Knoll¹, *Life Fellow, IEEE*

Abstract—The purpose of this work is to review the *state-of-the-art* LiDAR-based 3D object detection methods, datasets, and challenges. We describe novel data augmentation methods, sampling strategies, activation functions, attention mechanisms, and regularization methods. Furthermore, we list recently introduced normalization methods, learning rate schedules and loss functions. Moreover, we also cover advantages and limitations of 10 novel autonomous driving datasets. We evaluate novel 3D object detectors on the *KITTI*, *nuScenes*, and *Waymo* dataset and show their accuracy, speed, and robustness. Finally, we mention the current challenges in 3D object detection in LiDAR point clouds and list some open issues.

I. INTRODUCTION

Autonomous driving (AD) is increasingly gaining attention worldwide and can lead to many advantages to the population. The potential of this technology is clear, and it is predicted that it will dramatically change the transportation sector. The application of robust 3D Object Detection in autonomous driving is vital. In this context, robustness means to cope with detection errors (e.g., occlusions) and erroneous input (e.g., sensor noise). Sometimes, the environment is not optimal for detecting objects (e.g., in rain, snow, fog or bright sunlight). Robust detection stands for a good generalization to detect unseen objects in a different environment.

Besides, infrastructure sensors (like the ones in the *Providentia* [1] system) support the perception of the environment, improve traffic safety, and offer higher robustness and performance through different mounting positions. Multiple view points help to better detect objects in 3D (position, orientation, and size), and to increase the robustness against sunlight, snow, dust, and fog.

The aim of this survey paper is to provide an overview of novel 3D object detection methods and tricks. The best LiDAR-based 3D object detection algorithm on the *KITTI* dataset is 68.63% more accurate (on the 3D car class) than the best camera-only 3D object detection algorithm on that dataset. At night, the LiDAR performs better and is able to detect objects within a range of 50-60 m (Ouster OS1-64 gen. 2 LiDAR).

Our contribution is partitioned into the following:

- We provide *state-of-the-art* LiDAR-based 3D object detectors and show their accuracy, speed, and robustness.

The authors are with the Department of Informatics, Technical University of Munich (TUM), Garching, Germany.
E-mail: walter.zimmer@cs.tum.edu, {[ercelik](mailto:ercelik@in.tum.de), [zhou](mailto:zhou@in.tum.de), [diaz](mailto:diaz@in.tum.de), [knoll](mailto:knoll@in.tum.de)}

- Furthermore, we describe novel data augmentation methods, sampling strategies, activation functions, attention mechanisms, and regularization methods.
- We list recently introduced normalization methods, learning rate schedules and loss functions.
- We compare the most important datasets in detail, list their advantages and limitations, and evaluate the *state-of-the-art* detectors on the *KITTI* [2], *nuScenes* [3], and *Waymo* [4] datasets in terms of *mean average precision* (mAP) and inference speed.
- Finally, we mention current challenges in 3D object detection in LiDAR point clouds, list some open issues and provide research directions for the future.

II. RELATED WORK

[5] provides a fundamental analysis of 3D object detection stages. They compare several object detection network architectures, 3D box encodings, and evaluation metrics. [6] covers deep learning methods for several point cloud understanding tasks, including 3D object detection. [7] provides an overview of available datasets and evaluates the performance of object detection models. The authors summarize 3D object detection advancements for autonomous driving vehicles. Lastly, research gaps and future research directions are presented.

[8] separates LiDAR-based 3D object detection methods into 3D volume-based and projection-based methods, which consist of *birds-eye-view* (BEV) and *frontal view* (FV) projections. Furthermore, the authors list nine datasets for 3D object detection and autonomous driving. [9] describes the basic concepts of 3D bounding box encoding techniques, 3D data representation and sensing modalities. The authors introduce datasets that are helpful for deep learning-based 3D object detection projects. Furthermore, an in-depth systematic review of recent 3D object detection methods is shown as well as their limitations.

III. STATE-OF-THE-ART

3D Object Detection within point cloud data can be partitioned into a) point-based (input-wise permutation invariant) methods, b) voxel-based (grid-representation-based) methods, c) range-view-based methods and d) multi-view-based methods. Figure 1 shows a chronological view of the state-of-the-art detectors.

A. 3D Object Detection Architectures

We group novel architectures into one-stage architectures (*PointPillars*, *3DSSD*, *SA-SSD*, *CIA-SSD* and *SE-SSD*)

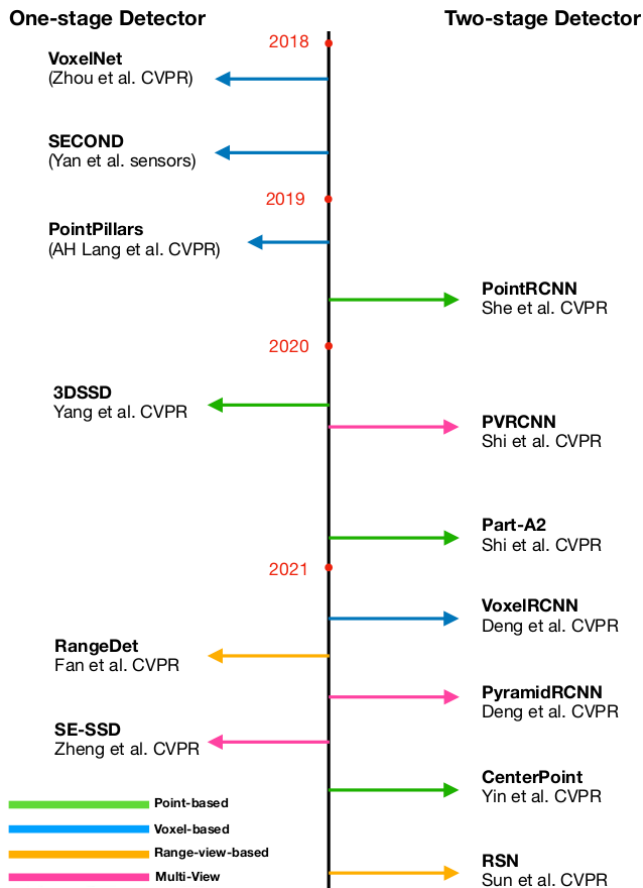


Fig. 1. Overview of the state-of-the-art detectors.

and two-stage architectures (*PV-RCNN*, *PV-RCNN++*, and *CenterPoint* and list their advantages, limitations and possible improvements.

PointPillars. This architecture is a point and voxel based method. First, the point cloud is divided into grids in the x - y coordinates, creating a set of pillars. Each point in the cloud (x , y , z , reflectance) is converted from a 4D vector into a 9D vector. The points are extended with the distance to the arithmetic mean of the pillar and the distance to the center of the pillar. Internally, *PointNet* is used to extract dense robust features from sparse *pillars*. Each 2D voxel cell (*pillar*) is represented by a fixed-size feature vector. 2D voxelization grids are more computationally efficient than those detection networks that take 3D voxelization grids as input. The 2D voxelization representation doesn't suffer from scale ambiguity and occlusion [10]. The encoder learns the point cloud representations from the *pillars*. In the end, a 2D CNN (modified SSD) is applied for joint bounding box and classification prediction. The end-to-end trainable *PointPillars* network runs at 62 Hz using *TensorRT*. *TensorRT* is a library for optimized GPU inference. [11] shows that switching to *TensorRT* increases inference speedup from 42.4 Hz to 62 Hz). It is highly efficient, because all key operations can be formulated as

2D convolutions that are extremely efficient to compute on a GPU. One limitation is that sometimes pedestrians and cyclists are misclassified. Pedestrians are confused with narrow vertical features of the environment such as poles or tree trunks. Cyclist are often misclassified because they are not much represented in the dataset.

3DSSD. *3DSSD* [12] (a single-stage 3D detection architecture) optimizes the inference speed while keeping the accuracy of two-stage detectors without voxel usage. The raw point-based 3D detectors first sample key points and extract features using set abstraction modules. To cover all objects in the scene, it's necessary to obtain raw point features from the extracted features of the key points. Since this process slows down the inference, the authors propose a new point sampling method, that uses the feature distance metric for key point sampling in addition to the Euclidean distance. This helps to increase the recall of the points and to balance positive and negative point samples. They also add a shift to the sampled candidate points before regression to have a better center prediction since the network is anchor-free. Additionally, the ground truth classification scores are changed with a value related to the distance of the candidate points to the surfaces of the ground truth bounding box, and therefore provides more realistic goals and further enhances the AP results. With all the extensions, the proposed architecture can reach an inference speed of 25 Hz. One downside of 3DSSD is its single-stage architecture that limits the 3D AP to 79.57% for the Car class (moderate difficulty).

SA-SSD. *SA-SSD* [13] (a single-stage 3D object detector) utilizes 3D convolutions on voxels as a backbone and 2D convolutions for the refinement and regression of the final 3D bounding boxes. The contribution of the architecture mainly arises from the proposed auxiliary network, that applies foreground segmentation and center estimation tasks on the interpolated LiDAR points and features. An additional segmentation task increases the sensitivity of the backbone to the boundaries of the objects. The center estimation enhances the relation of object points by learning to estimate object-specific centers from point-specific features. They also propose a feature map warping module to relieve the inconsistency between the positions of the refined anchors¹ and the classification scores by calculating classification as a consensus of object-related regions from the regression branch. With the given extensions and the ability of taking the auxiliary network out during inference, the *SA-SSD* network reaches 25 Hz, while improving the best scores of the single-stage detectors and reaching the accuracy of two-stage detectors.

¹Anchor boxes are essential if the input suffers from scale ambiguity, like natural images where the dimensions of the objects appearing in images depend on their distances to the camera, or when the network is required to detect different class objects (cars, bicycles and pedestrians) with different dimensions.

CIA-SSD. [14] proposed a new single-stage detector. Their contribution is a light-weight spatial-semantic *feature aggregation* module, that adaptively fuses high-level abstract semantic features and low-level spatial features for more accurate bounding box regression and classification. An *IoU-aware* confidence rectification module alleviates the misalignment between the localization accuracy and classification confidence. A distance-variant *IoU-weighted* NMS smooths the regressions and avoids redundant predictions.

PV-RCNN. Recently, [15] introduced the *PV-RCNN* architecture that integrates two point cloud feature learning strategies, a voxel-based and *PointNet*-based network. The goal is to use the best methods from both of them. First, a small set of key points is sampled from the point cloud to encode the whole scene. In parallel, it creates voxels from the raw point cloud and then encodes these 3D voxels to key points. The proposed *Voxel Set Abstraction* module incorporates the multi-scale voxel features into the nearby key points with the predefined balls and radii. The result is a key point feature vector containing information about the sampled key point and the 3D sparse convolutions. The key point features are reweighted by the *Predicted key point Weighting* (PKW) module. The motivation behind the reweighting is that foreground key points should contribute more to the accurate refinement of the proposals than the background key points. To group the key point features for proposal refinement, the authors propose the *RoI-grid pooling* via *Set Abstraction* (SA). Here, the key point features are aggregated (abstracted) to *RoI* grid points. In the end, a 3D bounding box refinement is taking place.

PV-RCNN++. *PV-RCNN++* [16] (a two-stage detection network) further improves the accuracy (+4% mAP on *Waymo*) and inference speed (+6 FPS² on *Waymo*). The authors proposed a new sampling strategy (*sectorized proposal-centric* strategy) to produce representative key points. It concentrates the limited key points to be around 3D proposals to encode more effective features for scene encoding and proposal refinement. Furthermore, a novel local feature aggregation method (*VectorPool* aggregation) was introduced. It replaces the previous set abstraction (SA) method by aggregating local point cloud features. The new method consumes less resources, and can handle large number of points.

CenterPoint [17]. In a first stage, the centers of objects are found, from which other attributes such as size, orientation, and velocity are estimated by different regression heads. By considering 3D objects as rotational-invariant points, this network is more flexible in the prediction of rotated objects with higher accuracy than anchor-based detectors. In a second stage, other point features, coming from the face

centers of the initial bounding box estimate, are condensed into a vector and fed into a MLP for prediction of confidence scores as well as for further refinement of the box attributes. The detection head is implemented as a heatmap in BEV. In this manner, every local peak of such heatmap corresponds to the center of a detected object. After processing the input point cloud for feature extraction, the result is a map-view, that is passed to a 2D CNN detection head to produce a K-channel heatmap – one channel for each K class respectively – with peaks at the center location of the detected object. In the training phase, the centers of ground truth bounding boxes are projected into a map-view so that the detection head can target a 2D Gaussian around these projections using a focal loss. A trained model on *nuScenes* with a *PointPillars* backbone runs at 31 FPS.

SE-SSD. [18] proposes a one-stage approach and adopts a single-stage detection architecture consisting of a teacher & student model. Both models share the same structure that resembles the CIA-SSD [14] architecture. The teacher model makes predictions based on the original point cloud, these are transformed into "soft targets" used to train the student. The student model makes predictions based on especially augmented data and is trained on the hard and soft targets via a *Consistency Loss* as well as a new *Orientation Aware Distance-IoU* loss (ODIoU). This loss emphasizes the orientation of the bounding boxes as well as the alignment of the center points. The authors propose a new *Shape-Aware Data Augmentation* scheme that performs dropout, swapping and sparsification of points. This data augmentation is applied to the point cloud data the student model is trained on. In this way, by using both a teacher and a student single-stage object detector, the framework can boost the precision of the detector significantly without incurring extra computation during the inference. At the same time, it also inherits the advantages of one-stage methods, namely the speed.

B. Data Augmentation

The goal of data augmentation is to improve generalization and make networks to become invariant with respect to rotation, translation, and natural changes in point clouds. Randomness of these data augmentation methods increases the generalization ability. Recently, it has become common to apply these data augmentation methods individually to bounding boxes of objects, which are called object-level data augmentation methods. Some common point cloud data augmentations are rotation around the vertical z-axis, flipping with respect to x-z and y-z planes, sparsification, additive *Gaussian noise*, *frustum dropout*, and scene augmentation with object point cloud frames. *AutoAugment* [19] formulates the sequential discrete decision problem as a *Reinforcement Learning* (RL) problem. The goal is to learn data augmentation strategies for object detection. It is shown that using the data augmentation policies generated by the RL agent outperforms the random data augmentation approach. Most of these data augmentation methods can either be applied at the scene-level or at object-level. [20] proposed a method

²On the *KITTI* dataset, the inference speed has improved from 13 to 17 FPS (+4 FPS).

that automates the design of data augmentation policies using an evolutionary search algorithm. Recently, *PointCutMix* [21] was proposed that finds the optimal assignment between two point clouds and generates new training data by replacing the points in one sample with their optimal assigned pairs.

C. Sampling Strategies

Farthest-Point-Sampling (FPS) [22] samples key points in different sectors to keep the uniformly distributed property of key points while accelerating the key point sampling process. It is not the optimal strategy because of quadratic complexity. The *sectorized proposal-centric key point sampling strategy* proposed in [16] concentrates the limited key points to be around 3D proposals to encode more effective features for scene encoding and proposal refinement. This strategy is much more efficient and can handle large-scale 3D scenes with million of points efficiently.

D. Activation Functions

The choice of activation functions is unavoidable when we design neural networks. They are introduced to ensure the non-linearity of the network, because the expressive power of linear models is usually insufficient, and can not satisfy the needs of various tasks in artificial intelligence. In neural networks, we usually add activation functions after each affine transformation to project and then keep activated features. *Swish* [23] was proposed in 2017, its definition is $Swish(x) = x * Sigmoid(x)$. The curve of *Swish* is similar to *ReLU*, but its derivative is continuous at 0 and its performance is slightly better. In the meanwhile, it brings higher computation cost. To reduce the computation cost, *Hard Swish* [24] uses *ReLU6* to approximate the *Swish* function. Its definition is $HardSwish(x) = x \frac{ReLU6(x+3)}{6}$. In comparison to *Swish*, its computation cost is much cheaper, so that it recently is getting high popularity.

E. Attention Mechanisms

Attention mechanisms are specially-designed parts of neural networks that learn to put emphasis on particularly important regions in the data. 3D Object Detection is one of the tasks where these mechanisms have recently been applied to. Especially, *attention mechanisms* can be considered to strengthen the robustness of voxel feature learning. *TANet* [25] presents a *triple attention* module embedded in the voxel feature extraction process and combines it with a proposed *cascaded refinement network*. Li *et al.* [26] implemented a novel feature pooling module (*ACA module*), to obtain a *perspective-invariant* prediction head. The attention mechanism is introduced to weight the contribution of each perspective differently. Existing IoU prediction methods suffer from the perspective variance. The proposed *ACA module* aggregates a local point cloud feature from each perspective of eight corners in the bounding box and adaptively weights the contribution of each perspective with a novel *perspective-wise* and *channel-wise* attention mechanism. Compared to the models that treat each corner and channel equally, the performance (mAP) increases slightly when the extracted features

are re-weighted by the *perspective-wise* and *channel-wise* attention. *CenterNet3D* [27] proposed an auxiliary *corner attention module* to enforce the CNN backbone to pay more attention to object boundaries, and to learn discriminative corner features. This leads to more accurate 3D bounding boxes. The corner attention module is implemented with a *corner classification module* that leads to a higher performance (+1-2% AP) without extra cost.

F. Regularization Strategies

Regularization strategies are designed to reduce the test error of a machine learning algorithm, possible at the expense of increased training error. While *Dropout* [28] is the most famous regularization method in the field of deep learning, many new methods were proposed recently. *DropBlock* [29] is a structured form of *Dropout* directed at regularizing convolutional networks. In *DropBlock*, units in a continuous region of a feature map are dropped together. As *DropBlock* discards features in a correlated area, the networks must look elsewhere for evidence to fit the data. The idea is to prevent co-adaptation, where the neural network becomes too reliant on particular areas, as this could be a symptom of overfitting.

G. Normalization Methods

The normalization method can help to generalize neural networks and accelerate the training process. *Batch Normalization* [30] was proposed to solve the *internal covariate shift* problem, which means the distribution of the layer inputs may get greatly changed after several weight updating iterations during the training process. It scales the output of the layer by standardizing the activated output of a node per mini-batch, and the standardized output is subject to the distribution with a mean of zero and a standard deviation of one. It also reduces the model dependence on the initial values of the parameters. *Filter Response Normalization* [31] is a combination of normalization and activation function. It performs normalization on each activation channel of each batch separately, during which the dependence of other batches is eliminated. *Attentive Normalization* [32] introduces the *attention mechanism* to the normalization technique. It re-weights the channel-wise affine transformation component by learning the instance-specific weights and computing the weighted sum of mixture for feature re-calibration.

H. Learning Rate Schedules

At the beginning of training, the weights of a model are randomly initialized where the model may become unstable with a large learning rate. To solve this problem, *Warmup* [33] chooses a relatively small learning rate at the beginning of the training process and then increases it a few epochs later. *Cosine Annealing* is widely applied in the training of 3D object detection models such as in *PV-RCNN* [34] and *CIA-SSD* [14]. It's a straightforward strategy that the learning rate decreases following a cosine function where the value declines slowly at the beginning and the end while drops rapidly in the middle. The property of cosine functions fits well with the learning rate and sometimes it

is also combined with warm restart. In warm restart, cosine annealing is repeated. The restarted learning rate depends on the parameters of the last iteration instead of a fixed value. *Warmup* and the cosine annealing strategy can also be combined [35].

I. Loss Functions

The default classification loss for object detection networks is the *cross-entropy loss* that is equal to the negative log likelihood (NLL) of the ground truth class. The main complication of the classification loss of object detection networks is to address the class imbalance between the background (negative) class and positive classes. Most of the potential bounding box candidates contain background scenes and not target objects. The two standard approaches to address this class imbalance are *hard negative mining* and *focal loss* [36]. In the *hard negative mining*, negative bounding boxes are sub-sampled per scene such that the number of negative boxes is at most three times higher. *Focal loss* adaptively adjusts the contributing weight of each example in the classification loss during the training process instead of explicitly sub-sampling the negative examples.

The *PointPillars* architecture combines three loss functions: localization loss (\mathcal{L}_{loc}), classification loss (\mathcal{L}_{cls}) and the direction loss (\mathcal{L}_{dir}).

$$\begin{aligned}\mathcal{L}_{loc} &= \sum_{b \in (x,y,z,w,l,h,\theta)} SmoothL1(\Delta b) \\ \mathcal{L}_{cls} &= -\alpha_a(1-p^a)^\gamma \log(p^a) \\ \mathcal{L}_{dir} &= SmoothL1(\sin(\theta_p - \theta_t)) \\ \mathcal{L}_{total} &= \frac{1}{N_{pos}}(\beta_{loc}\mathcal{L}_{loc} + \beta_{cls}\mathcal{L}_{cls} + \beta_{dir}\mathcal{L}_{dir})\end{aligned}$$

Focal loss is used as object classification loss (\mathcal{L}_{cls}), and softmax classification loss is used as direction loss (\mathcal{L}_{dir}). The total loss is a weighted sum of all three losses.

[37] proposed an *IoU loss* for rotated 3D bounding boxes (that considers the rotation in two directions), since an axis-aligned box is not suitable for representing target objects in 3D. This loss can be applied for both axis-aligned or rotated 2D/3D objects. The IoU between two 3D bounding boxes (d and g) that are rotated in one direction (e.g. yaw) can be calculated as follows:

$$IoU_{3D} = \frac{Area_{overlap} \cdot h_{overlap}}{Area_g \cdot h_g + Area_d \cdot h_d - Area_{overlap} \cdot h_{overlap}}$$

$h_{overlap}$ and h_{union} represent the intersection and union in the height direction. The IoU loss is then defined as $L_{IoU} = 1 - IoU$.

IV. AUTONOMOUS DRIVING DATASETS

Increasing availability of labeled datasets with millions of samples allows training models more accurately. A brief summary of some of the most well-known datasets and competitions is presented in Table I.

Datasets in autonomous driving contain multiple modalities to provide a robust and comprehensive understanding of the whole scene. Besides LiDAR information, most datasets

also provide corresponding RGB images, which are dense and contain more semantic features to compensate for the sparsity in LiDAR point clouds. The cost to obtain RGB images is also much lower than getting point clouds. Some datasets also provide radar information such as *nuScenes*. In comparison with the LiDAR sensor, radar has a farther scanning range and is less affected by extreme weather conditions. The first batch of the recently released *A9-Dataset* contains infrastructure sensor data (camera and LiDAR) from the A9 Highway (including an accident) and a large traffic intersection with labeled vehicles and vulnerable road users (VRUs).

V. EVALUATION

Table II shows the mAP and inference speed of the mentioned *state-of-the-art* methods (grouped by one-stage and two-stage detectors) on the *KITTI*, *nuScenes* and *Waymo Level 1* test set.

The AP on the *KITTI* test set was evaluated on the moderate 3D object difficulty of the car class using 40 recall positions (*R40*). The rotated *IoU* on the *KITTI* test set was set to a threshold of 0.7 for cars and 0.5 for pedestrian and cyclists. *PointPillars* runs at 62 FPS using *TensorRT* for GPU acceleration. Using the *PyTorch* pipeline, it runs at 42 FPS according to [11]. Evaluation on the *KITTI* test server was performed with a 1080TI GPU and an Intel i7 CPU (1 core @2.5 Ghz). The overall 3D mAP on the *Waymo Level 1* dataset was evaluated at an intersection-over-union (IoU) of 0.7. The detection region was set to 150 x 150 m ($[-75.2\text{ m}, 75.2\text{ m}]$ for the X and Y axis) and $[-2\text{ m}, 4\text{ m}]$ for the Z axis. According to [46] the inference speed of *PV-RCNN* on the *Waymo* dataset is about 300 ms (4 FPS).

VI. CHALLENGES AND OPEN FIELDS

Multi-frame 3D Object Detection. Current state-of-the-art 3D object detection models mainly rely on LiDAR point clouds or its combination with other data types for reaching the best results. The dynamic nature of traffic causes occlusions that occur in fixed time intervals. LiDAR point cloud sparsity is another problem that hinders perfect 3D detection. The 3D object detection community has recently started to consider using data history with the release of large datasets such as *nuScenes* [3] and *Waymo Open Dataset* [4] in addition to the *KITTI Raw and Tracking* datasets [2]. We introduce the most recent multi-frame 3D object detection studies in four primary categories with some obvious overlaps: (i) implicit multi-frame data processing, (ii) scene-level feature aggregation, (iii) object-level feature aggregation, and (iv) aggregation with attention mechanisms.

We define implicit multi-frame approaches as not explicitly using learning-based methods for aggregating features from multiple frames. *CenterPoint* [17] estimates center offsets (velocities) between two successive frames for 3D detection and tracking. [47] aggregates occupancy maps using Bayesian filtering through sequential point cloud sweeps. The method fuses the temporal occupancy map with the

TABLE I
AUTONOMOUS DRIVING DATASETS THAT ARE USED FOR 3D OBJECT DETECTION.

Name	Year	PC frames	RGB images	Classes	Diff. weather/time	Obj./frame	Location
KITTI [2]	2012	15.4k	15k	8	No/No	13	Karlsruhe (Germany)
nuScenes[3]	2019	400k	1.4M	23	Yes/Yes	35	Boston (USA), Singapore (SG)
Waymo [4]	2019	200k	1M	4	Yes/Yes	60	SF , Phoenix, Mt. View (USA)
ArgoVerse [38]	2019	44k	490k	15	Yes/Yes	45	Pittsburgh, Miami (USA)
Lyft Lvl. 5[39]	2020	323k	46k	9	No/No	28	Palo Alto (USA)
A2D2 [40]	2020	41k	41k	38	Yes/No	-	Ingolstadt (Germany)
LIBRE [41]	2020	-	-	-	Yes/Yes	-	Nagoya (Japan)
ONCE [42]	2021	16k	16k	5	Yes/Yes	26	China
IPS300+ [43]	2021	1.25k	2.5k	8	No	320	China
A9-Dataset [44]	2022	17.4k	68.4k	10	Yes/Yes	34	Munich (Germany)

TABLE II
COMPARISON OF THE *state-of-the-art* METHODS ON THE *KITTI* (3D CAR, MODERATE), *nuScenes* AND *Waymo Level 1* TEST SET.

Method	KITTI			nuScenes		Waymo	
	Year	3D mAP	FPS (ms)	NDS	FPS (ms)	3D mAP	FPS (ms)
PointPillars [11]	2019	74.31%	62 (16 ms)	0.44	62 (16 ms)	45.52%	62 (16 ms)
3DSSD [12]	2020	79.57%	26 (38ms)	-	-	-	-
SA-SSD [13]	2020	79.79%	25 (40ms)	-	-	61.48%	25 (40ms)
CIA-SSD [45]	2021	80.28%	33 (30 ms)	-	-	-	-
SE-SSD [18]	2021	83.73%	33 (30 ms)	-	-	-	-
PV-RCNN [15]	2020	81.43%	13 (80 ms)	-	-	70.30%	4 (300 ms)
PV-RCNN++ [16]	2021	81.88%	16 (60 ms)	-	-	74.81%	10 (100 ms)
CenterPoint [17]	2021	74.87%	20 (51 ms)	0.71	16 (60 ms)	79.25%	11 (89 ms)

point cloud feature maps. In [48], object proposals are generated based on temporal occupancy maps, odometry data, and Kalman filter-based motion estimations. [49] aggregates two consecutive feature maps using odometry-based flow estimation. [50] combines point clouds from multiple frames by tagging the points with timestamps, which are aligned using the ego-motion compensation and used as the input.

The scene-level multi-frame feature map aggregation approach aims to construct a richer temporal feature map to enhance 3D object detection accuracy. [51] utilizes 3D convolutional layers to process multi-frame BEV voxel features across time for 3D detection, tracking, motion forecasting tasks. Instead of 3D convolutions, convolutional LSTMs are used for spatio-temporal BEV feature map aggregation in [52] and [53]. Similarly, [54] proposes a custom convolutional LSTM for multi-frame feature map fusion in addition to the ego-motion compensation for positional feature offsets. *SDP-Net* [55] generates an aggregated feature map from sequential BEV features based on flow and ego-motion estimations. Detection and tracking results are obtained from the final temporal feature map.

Object-level feature aggregation in time aims at obtaining better object representations using object feature vectors from preceding detections. [56] uses recurrent layers and 2D object tracking IDs to match and aggregate object features in time. [57] generates multi-frame object proposal features from the object feature vectors in the memory using a non-local attention block for aligning features. [58] processes LiDAR point cloud sequences for automatic labeling. A 3D multi-object tracker is used to match single-frame detections. A final stage refines the 3D bounding boxes after merging

points of matched objects in time. [59] combines point clouds from multiple frames using the calibration information and generates proposals from the sizeable combined point clouds. A spatio-temporal GNN further refines the proposals.

Multi-frame attention mechanisms have taken a growing interest in 3D object detection. [57] applies non-local attention blocks to consecutive object features to align and generate temporal object features. [60] and [61] use spatial and temporal transformer attention modules to align the temporal feature maps generated with a convolutional GRU. Similarly, [62] applies a spatio-temporal transformer module to obtain a temporally-enhanced feature map, which a detection head takes as input.

Multi-frame studies show that feature aggregation through successive frames enhances the 3D detection accuracy. However, alignment remains an issue. Methods so far have used ego-motion, calibration, or tracking-based alignment methods. Also, convolutional recurrent networks' filters expectedly account for such offsets between sequential feature maps. Attention mechanisms have stood out as a promising solution to the alignment issue. Even though many graph network-based 3D detection methods exist, direct feature alignment with graph networks for 3D detection is surprisingly a new open field. Therefore, transformer attention mechanisms and temporal graph representations remain prospective fields for multi-frame 3D detection.

Imbalanced datasets and point cloud sparsity. Class imbalance like in the *KITTI* dataset (100k cars, 25k pedestrians, 7k cyclists) leads to a bad detection of pedestrians and cyclists. *nuScenes* is about two times larger than the *KITTI* dataset and contains more scenarios

at different times (day, night) and weather conditions (sunny, rainy, cloudy). Each scene has a full 360 degree field-of-view. However, the number points per scene in *nuScenes* is only a third of *KITTI*. Detecting small objects in sparse and irregular LiDAR point clouds is difficult. The *Waymo Open* dataset is one of the largest open source 3D object detection datasets so far in autonomous driving. One obvious drawback of the *Waymo* dataset is, that it only provides 4 classes.

Cross-dataset domain adaptation [63]. After training an algorithm on a large publicly available dataset, one often wants to apply this algorithm on own data. The problem here is often, that different sensors and mounting positions are used, so that the trained model is not able to detect objects. Car sizes for example are varying across different geographic regions. Most of the novel datasets contain cars that were only observed from one side. The depth of the bounding boxes must be estimated based on gathered experience during training.

Data annotation. Accurate data annotation in 3D space is a very tedious task and is often outsourced to expert annotators. Open source 3D annotation tools like [64] automate the annotation process so that a large number of annotations can be created in a short time. After annotating your own data and retraining your model on that annotated data, you should receive better detection and classification results.

VII. CONCLUSION

This paper has presented a survey of novel methods, *state-of-the-art* 3D object detection architectures, and their comparison (in terms of mean average accuracy and inference speed). We hope this will help practitioners to choose an appropriate method when deploying object detection algorithms in the real world. We have also identified some new techniques for improving speed without sacrificing much accuracy, e.g. by using *TensorRT* for GPU acceleration. The attention mechanism is recently being applied also in 3D object detection to further increase the mAP. In detail, we described novel data augmentation methods, sampling strategies, activation functions, attention mechanisms, and regularization methods. We listed recently introduced normalization methods, optimization methods, learning rate schedules, loss functions, and evaluation metrics. Finally, potential research directions, challenges and issues of 3D object detection methods in point clouds were given.

ACKNOWLEDGMENT

This work was funded by the Federal Ministry of Transport and Digital Infrastructure, Germany as part of the *Providentia++* research project (Grant Number: 01MM19008A). The authors would like to express their gratitude to the funding agency and to the numerous students at TUM for technical editing, language editing, and proofreading.

REFERENCES

- [1] A. Krämmer, C. Schöller, D. Gulati, and A. Knoll, "Providentia-a large scale sensing system for the assistance of autonomous vehicles," *arXiv preprint arXiv:1906.06789*, 2019.
- [2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," p. 6, 2013.
- [3] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A Multimodal Dataset for Autonomous Driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 621–11 631.
- [4] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.
- [5] Y. Wu, Y. Wang, S. Zhang, and H. Ogai, "Deep 3D Object Detection Networks Using LiDAR Data: A Review," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1152–1171, Jan. 2021.
- [6] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.
- [7] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A Survey on 3D Object Detection Methods for Autonomous Driving Applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, Oct. 2019.
- [8] Y. Huang and Y. Chen, "Survey of State-of-Art Autonomous Driving Technologies with Deep Learning," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, Dec. 2020, pp. 221–228.
- [9] M. M. Rahman, Y. Tan, J. Xue, and K. Lu, "Notice of Violation of IEEE Publication Principles: Recent Advances in 3D Object Detection in the Era of Deep Neural Networks: A Survey," *IEEE Transactions on Image Processing*, vol. 29, pp. 2947–2962, 2020.
- [10] M. Sanatkar, "Lidar 3d Object Detection Methods," <https://towardsdatascience.com/lidar-3d-object-detection-methods-f34cf3227aea>, Jun. 2020.
- [11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast Encoders for Object Detection From Point Clouds," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, Jun. 2019, pp. 12 689–12 697.
- [12] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3dssd: Point-based 3d single stage object detector," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 040–11 048.
- [13] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3d object detection from point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 873–11 882.
- [14] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, "CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud," Tech. Rep., 2021. [Online]. Available: <https://github.com/Vegeta2020/CIA-SSD>.
- [15] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection," 2020, pp. 10 529–10 538.
- [16] S. Shi, L. Jiang, J. Deng, Z. Wang, C. Guo, J. Shi, X. Wang, and H. Li, "Pv-rnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection," *arXiv preprint arXiv:2102.00463*, 2021.
- [17] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 11 784–11 793.
- [18] W. Zheng, W. Tang, L. Jiang, and C.-W. Fu, "Se-ssd: Self-ensembling single-stage object detector from point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 494–14 503.
- [19] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama, "Faster augmentation: Learning augmentation strategies using backpropagation," in *European Conference on Computer Vision*. Springer, 2020, pp. 1–16.
- [20] "Improving 3D Object Detection through Progressive Population Based Augmentation," <https://deepai.org/publication/improving-3d-object-detection-through-progressive-population-based-augmentation>, Apr. 2020.

- [21] J. Zhang, L. Chen, B. Ouyang, B. Liu, J. Zhu, Y. Chen, Y. Meng, and D. Wu, "Pointcutmix: Regularization strategy for point cloud classification," *arXiv preprint arXiv:2101.01461*, 2021.
- [22] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, 1997.
- [23] P. Ramachandran, B. Zoph, and Q. V. Le Google Brain, "SWISH: A SELF-GATED ACTIVATION FUNCTION," Tech. Rep.
- [24] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song, "The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks," nov 2019. [Online]. Available: <http://arxiv.org/abs/1911.07135>
- [25] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "Tanet: Robust 3d object detection from point clouds with triple attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 11 677–11 684.
- [26] J. Li, S. Luo, Z. Zhu, H. Dai, A. S. Krylov, Y. Ding, and L. Shao, "3d iou-net: Iou guided 3d object detector for point clouds," *arXiv preprint arXiv:2004.04962*, 2020.
- [27] G. Wang, B. Tian, Y. Ai, T. Xu, L. Chen, and D. Cao, "Centernet3d: An anchor free object detector for autonomous driving," *arXiv preprint arXiv:2007.07214*, 2020.
- [28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [29] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," *arXiv preprint arXiv:1810.12890*, 2018.
- [30] S. Ioffe and C. Szegedy, "Batch Normalization," Tech. Rep., 2015.
- [31] S. Singh, S. Krishnan, and G. Research, "Filter Response Normalization Layer: Eliminating Batch Dependence in the Training of Deep Neural Networks," Tech. Rep.
- [32] X. Li, W. Sun, and T. Wu, "Attentive Normalization," Tech. Rep. [Online]. Available: <https://github.com/iVMCL/>
- [33] I. Loshchilov and F. Hutter, "SGDR: STOCHASTIC GRADIENT DESCENT WITH WARM RESTARTS," Tech. Rep. [Online]. Available: <https://github.com/loshchil/SGDR>
- [34] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection," Tech. Rep.
- [35] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts," *arXiv:1608.03983 [cs, math]*, May 2017.
- [36] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
- [37] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "Iou loss for 2d/3d object detection," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 85–94.
- [38] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8748–8757.
- [39] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Igloukov, and P. Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," *arXiv preprint arXiv:2006.14480*, 2020.
- [40] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühlegg, S. Dorn *et al.*, "A2d2: Audi autonomous driving dataset," *arXiv preprint arXiv:2004.06320*, 2020.
- [41] A. Carballo, J. Lambert, A. Monroy, D. Wong, P. Narksri, Y. Kit-sukawa, E. Takeuchi, S. Kato, and K. Takeda, "Libre: The multiple 3d lidar dataset," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1094–1101.
- [42] J. Mao, M. Niu, C. Jiang, H. Liang, J. Chen, X. Liang, Y. Li, C. Ye, W. Zhang, Z. Li *et al.*, "One million scenes for autonomous driving: Once dataset," *arXiv preprint arXiv:2106.11037*, 2021.
- [43] H. Wang, X. Zhang, J. Li, Z. Li, L. Yang, S. Pan, and Y. Deng, "Ips300+: a challenging multimodal dataset for intersection perception system," *arXiv preprint arXiv:2106.02781*, 2021.
- [44] C. Cress, W. Zimmer, L. Strand, M. Fortkord, S. Dai, V. Lakshminarasimhan, and A. Knoll, "A9-dataset: Multi-sensor infrastructure-based dataset for mobility research," *arXiv preprint arXiv*, 2022.
- [45] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, "Cia-ssd: Confident iou-aware single-stage object detector from point cloud," *arXiv preprint arXiv:2012.03015*, 2020.
- [46] S. Shi, "Execution speed of PVRCNN on waymo dataset per point cloud? · Issue #354 · open-mmlab/OpenPCDet," <https://github.com/open-mmlab/OpenPCDet/issues/354>, Nov. 2020.
- [47] P. Hu, J. Ziglar, D. Held, and D. Ramanan, "What you see is what you get: Exploiting visibility for 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 001–11 009.
- [48] J. Sun, Y. Xie, S. Zhang, L. Chen, G. Zhang, H. Bao, and X. Zhou, "You don't only look once: Constructing spatial-temporal memory for integrated 3d object detection and tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3185–3194.
- [49] Y. Murhij and D. Yudin, "Real-time 3d object detection using feature map flow," *arXiv preprint arXiv:2106.14101*, 2021.
- [50] A. Piergiovanni, V. Casser, M. S. Ryoo, and A. Angelova, "4d-net for learned multi-modal alignment," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 435–15 445.
- [51] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 3569–3577.
- [52] A. El Sallab, I. Sobh, M. Zidan, M. Zahran, and S. Abdelkarim, "Yolo4d: A spatio-temporal approach for real-time multi-object detection and classification from lidar point clouds," 2018.
- [53] S. McCrae and A. Zakhori, "3d object detection for autonomous driving using temporal lidar data," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 2661–2665.
- [54] R. Huang, W. Zhang, A. Kundu, C. Pantofaru, D. A. Ross, T. Funkhouser, and A. Fathi, "An lstm approach to temporal 3d object detection in lidar point clouds," *arXiv preprint arXiv:2007.12392*, 2020.
- [55] Y. Zhang, Y. Ye, Z. Xiang, and J. Gu, "Sdp-net: Scene flow based real-time object detection and prediction from sequential 3d point clouds," in *Proceedings of the Asian Conference on Computer Vision*, 2020.
- [56] E. Erçelik, E. Yurtsever, and A. Knoll, "Temp-frustum net: 3d object detection with temporal fusion," 2021.
- [57] Z. Yang, Y. Zhou, Z. Chen, and N. Ngiam, "3d-man: 3d multi-frame attention network for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1863–1872.
- [58] C. R. Qi, Y. Zhou, M. Najibi, P. Sun, K. Vo, B. Deng, and D. Anguelov, "Offboard 3d object detection from point cloud sequences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6134–6144.
- [59] Z. Xiong, H. Ma, Y. Wang, T. Hu, and Q. Liao, "Lidar-based 3d video object detection with foreground context modeling and spatiotemporal graph reasoning," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 2994–3001.
- [60] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 495–11 504.
- [61] J. Yin, J. Shen, X. Gao, D. Crandall, and R. Yang, "Graph neural network and spatiotemporal transformer attention for 3d video object detection from point clouds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- [62] Z. Yuan, X. Song, L. Bai, Z. Wang, and W. Ouyang, "Temporal-channel transformer for 3d lidar-based video object detection for autonomous driving," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.
- [63] Y. Wang, X. Chen, Y. You, L. E. Li, B. Hariharan, M. Campbell, K. Q. Weinberger, and W.-L. Chao, "Train in germany, test in the usa: Making 3d object detectors generalize," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 713–11 723.
- [64] W. Zimmer, A. Rangesh, and M. Trivedi, "3D BAT: A Semi-Automatic, Web-Based 3D Annotation Toolbox for Full-Surround, Multi-Modal Data Streams," <http://arxiv.org/abs/1905.00525>, May 2019. [Online]. Available: <https://github.com/walzimmer/3d-bat>